Matrix powers algorithms for trust evaluation in PKI architectures

Jean-Guillaume Dumas[*] Hicham Hossayni[*][†]

December 19, 2011

Abstract

This paper deals with the evaluation of trust in a network of public-key infrastructures. We consider the PGP web of trust or the network of cross-certificates of trust anchors for website authentication. Different trust models have been proposed to interconnect the various PKI components in order to propagate the trust between them. In this paper we provide a simple model for trust and reputation management in PKI architectures, and a new polynomial algorithm using linear algebra to assess trust relationships in a network using different trust evaluation schemes.

1 Introduction

The principle of a Public Keys Infrastructure (PKI) is to establish (using certificates) a trust environment between network entities and thus guaranty some security of communications.

For example in a cross-certification PKI, an entity called Alice can establish a communication with another entity called Bob only after validating Bob's certificate. For this, Alice must verify the existence of a certification path between her trust anchor and Bob's certification authority (CA). This certificate validation policy imposes that each entity must have a complete trust in its trust anchors, and that this trust anchor has a direct or indirect relation with other CAs.

In fact, several risks exist in the current trust models if the trust anchors are not validated out-of-band by their users. Ellison and Schneier identified a risk of PKIs to be "Who do we trust, and for what?" which emphasizes the doubts about the trust relationship between the different PKI components [3]. Several incidents, including the one in which VeriSign issued to an fraudulent two certificates associated with Microsoft [5], or even the recent fraudulent certificates

[*]Laboratoire J. Kuntzmann, Université de Grenoble. 51, rue des Mathématiques, umr CNRS 5224, bp 53X, F38041 Grenoble, France, `Jean-Guillaume.Dumas@imag.fr`.

[†]CEA/DRT LETI/DCIS, 17 rue des martyrs, 38054 Grenoble, France, `Hicham.Hossayni@cea.fr`.

for Google emitted by DigiNotar [1], confirms that a global evaluation of trust for the trust anchors might be a solution to assess a respective global degree of trust. In e.g.

[6, 7, 8] algorithms are proposed to quantify the trust relationship between two entities in a network, using transitivity. Some of them evaluate trust throughout a single path, while others consider more than one path to give a better approximation of trust between entities. However to the best of our knowledge they are restricted to simple network trees. Another approach would be to use some fully trusted keys or authorities, like the Sovereign Keys or the Convergence projects[1].

In this paper we choose the first approach and use transitivity to approximate global levels of trust, efficiently. Our idea is to use the powers of the incidence matrix (used e.g. to verify the graph connexity or to compute the number of [bounded] paths between nodes). The approach is similar to that used also e.g. for community detection in graphs [4] and we use it to produce a centralized or distributed quantification of trust in a network. The complexity of this algorithm is $O(n^3 \cdot \varphi \cdot \ell)$ in the worst case, polynomial in n, the number of entities (nodes of the graph), φ, the number of trust relationships (edges), and ℓ, the size of the longest path between entities. For instance the algorithm proposed in [7] worked only for directed acyclic graphs (DAG) and required the approximate resolution of the Bounded Disjoint Paths problem, known to be NP-Hard [12]. In case of DAGs the complexity of our algorithm even reduces to $O(n \cdot \varphi \cdot \ell)$.

The aim of our algorithm is the evaluation of trust using all existing (bounded) trust paths between entities as a preliminary to any exchanges between PKIs. This can give a precise evaluation of trust, and optimize the certificate validation time. The algorithm can also be adapted (under condition) to different trust metrics.

We present different our chosen trust metric in section 2 and our algorithm in section 3. Then, we show that trust and reputation are complementary and propose a simple model for the trust management in PKI architectures, using both notions (trust & reputation).

2 Transitive trust metric

There are several schemes for evaluating the (transitive) trust in a network. Some presents the trust degree as a single value representing the probability that the expected action will happen.

Others include the *distrust* degree indicating the probability that the opposite of the expected action will happen [6]. More complete schemes can be introduced to evaluate trust: Jøsang [9] for instance introduced the Subjective Logic notion which expresses subjective beliefs about the truth of propositions with degrees of "uncertainty".

[1]https://www.eff.org/sovereign-keys, http://convergence.io

[7, 8] also introduced a quite similar scheme with a formal, semantics based, calculus of trust and applied it to public key infrastructures (PKI). In the next sections, we adopt the latter trust evaluation scheme: the idea is to represent trust by a triplet, (trust, distrust, uncertainty). Trust is the proportion of experiences proved, or believed, positive. Distrust is the proportion of experiences proved negative. Uncertainty is the proportion of experiences with unknown character.

Definition 1. *Let d be a trustor entity and e a trustee. Let m be the total number of encounters between d and e regarding an instanced expectancy in a given context. Let n (resp. l) be the number of positive (resp. negative) experiences among all encounters between d and e.*

- **The trust degree** *is defined as the frequency rate of the trustor's positive experience among all encounters with the trustee. That is, $td(d, e) = \frac{n}{m}$.*
- **The distrust degree***: similarly we have $dtd(d, e) = \frac{l}{m}$.*
- **The uncertainty***: denoted by ud is defined by: $ud = 1 - td - dtd$.*

In the following we will denote the trust relationship by $tr(a, b) =< td(a, b), dtd(a, b), ud(a, b) >$ or simply $tr(a, b) =< td(a, b), dtd(a, b) >$ since the uncertainty depends directly of the trust and distrust degrees.

In these definitions, the trust depends on the kind of expectancy, the context of the experiences, type of trust (trust in belief, trust in performance), ..., see e.g. [8]. For simplicity, we only consider in the next sections the above generic concept of trust.

2.1 Aggregation of trust

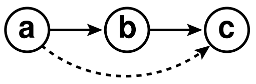
The main property we would like to express is *transitivity*. Indeed in that case keys trusted by many entities, themselves highly trusted, will induce a larger confidence. In the following we will consider a trust graph representing the trust relationships as triplets between entities in a network.

Definition 2. ***(Trust graph)*** *Let $\mathcal{T} \subset \mathbb{R}^3$ be a set of trust values. Let V be a set of entities of a trust network. Let E be a set of directed edges with weight in $\mathcal{T}$. Then $G = (V, E, \mathcal{T})$ is called a* trust graph *and there is an edge between two vertices whenever there exist a nonzero trust relationship between its entities.*

Next we define the transitivity over a path between entities and using parallel path between them as sequential and parallel aggregations. We first need to define a trust path:

Definition 3. ***(Trust path)*** *Let $G = (V, E, \mathcal{T})$ be a trust graph. A trust path between two entities $T_1 \in V$ and $T_n \in V$ is represented as the chain, $T_1 \xrightarrow{v_1} T_2 \xrightarrow{v_2} ...T_{n-1} \xrightarrow{v_{n-1}} T_n$, where T_i are entities in V and $v_i \in \mathcal{T}$ are respectively the trust degrees associated to each trust relation $(T_i \xrightarrow{v_i} T_{i+1}) \in E$.*

The need of the sequential aggregation is shown by the following example. Consider, as shown on figure 1, Alice trusting Bob with a certain degree, and Bob trusting Charlie with a certain trust degree. Now, if Alice wishes to communicate with Charlie, how can she evaluate her trust degree toward Charlie? For this, we use the sequential aggregation of trust to help Alice to make a decision, and that based on Bob's opinion about Charlie.



$\longrightarrow$: Direct trust relashionship

$\dashrightarrow$: Indirect (sequentially aggregated) trust relashionship

Figure 1: Simple sequential trust aggregation

Definition 4. ***(Sequential aggregation of trust)*** *Let $G = (V, E, \mathcal{T})$ be a trust graph. Let a,b and c be three entities in V and $tr(a, b) \in \mathcal{T}$, $tr(b, c) \in \mathcal{T}$ be respectively the trust degrees associated to the entity pairs (a, b) and (b, c). The* sequential aggregation of trust *between a and c is a function f, that calculates the trust degree over the trust path $a \to b \to c$. It is defined by :*

$$\begin{aligned} f : \mathcal{T} \times \mathcal{T} \to \mathcal{T} \text{ with } \quad & f(tr(a,b), tr(b,c)) = tr_f(a,c) =< td_f(a,c), dtd_f(a,c) > \\ \text{where } \ td_f(a,c) \quad & = td(a,b).td(b,c) + dtd(a,b).dtd(b,c) \\ dtd_f(a,c) \quad & = dtd(a,b).td(b,c) + td(a,b).dtd(b,c) \end{aligned}$$

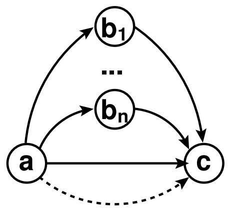
This definition of f the sequential aggregation function is given by [7, Theorem UT-1]. This sequential aggregation function can be applied recursively to any tuple of values of $\mathcal{T}$, to evaluate the sequential aggregation of trust over any trust path with any length ≥ 2 as follows: $f(v_1, ..., v_n) = f(f(v_1, ..., v_{n-1}), v_n)$.

Now, the following definition of the parallel aggregation function can also be found in [8, § 7.2.2] and is illustrated on figure 2.

Definition 5. ***(Parallel aggregation of trust)*** *Let $G = (V, E, \mathcal{T})$ be a trust graph. Let $a, b_1, \ldots, b_n$, and c be entities in V and and $tr_i(a, c) \in \mathcal{T}$ be the trust degree over the trust path $a \to b_i \to c$ for all $i \in 1..n$. The parallel aggregation of trust is a function g, that calculates the trust degree associated to a set of disjoint trust paths connecting the entity a to the entity c. it is defined by*

$$\begin{aligned} & g : \mathcal{T}^n \to \mathcal{T} \text{ with } g([tr_1, tr_2, \ldots, tr_n](a,c)) = tr_g(a,c) =< td_g(a,c), dtd_g(a,c) > \\ & \quad \text{where } \ td_g(a,c) = 1 - \prod_{i=1..n} (1 - td_i) \text{ and } dtd_g(a,c) = \prod_{i=1..n} dtd_i \end{aligned}$$

Definition 6. ***(Trust evaluation)*** *Let $G(V, E, \mathcal{T})$ be a directed acyclic trust graph, and let a and b be two nodes in V. The trust evaluation between a and b is the trust aggregation over* all *paths connecting a to b. It is computed recursively by aggregating (evaluating) the trust between the entity a and the predecessors of b (except, potentially, a). Denote by $Pred(b)$ the predecessors of b and by p_i*



$\longrightarrow$: Direct trust relashionship

$\dashrightarrow$: Indirect (parallely aggregated) trust relashionship

Figure 2: Parallel aggregation of trust for multiple trust

the elements of $Pred(b) \setminus \{a\}$*. The* trust evaluation *between* a *and* b *consists in applying first the sequential aggregation over the paths* $a \rightarrow p_i \rightarrow b$ *and then the parallel aggregation to the results and* $(a \rightarrow b)$ *(if* $(a \rightarrow b) \in E$*).*

The graph theoretic method proposed by [7, §6.3] for evaluating trust between two nodes in a DAG requires the approximate solution of the Bounded Disjoint Paths problem, known to be NP-Hard [12]. We propose in algorithm 1 to remove the search for disjoint paths and then we obtain a polynomial time algorithm.

Algorithm 1 Recursive trust evaluation

```
Input $G = (V, E, \mathcal{T})$ a direct acyclic trust graph, A, Z two nodes of G.
Output Trust between A and Z
1: calculate $N = Pred(Z) \setminus \{A\}$;
2: For all $n_i \neq$ A in N aggregate(A, $n_i$, G);
3: use parallel aggregation to aggregate all paths from A to Z;
```

By storing the already evaluated relationships, this algorithm can e.g. compute the global trust in a graph with 1000 vertices and 250 252 edges in less than 58 seconds on a standard laptop. In the following, we propose to rewrite this algorithm in terms of linear algebra. Using sparse linear algebra the overall complexity will not change, and the analysis will be eased. Now if the graph is close to complete, the trust matrix will be close to dense and cache aware linear algebra algorithms will be more suited. Moreover, the linear algorithm will decompose the evaluation into converging iterations which could be stopped before exact convergence in order to get a good approximation faster. Then we will also deduce from this point of view a generalization to any directed graph.

3 Matrix Powers algorithm

In this section, we propose a new algorithm for evaluating trust in a network using the powers of the matrix of trust. This algorithm uses techniques from graph connexity

and communicability in networks [4].

Our matrix powers algorithm can be implemented with different trust propagation schemes under one necessary condition: the transitivity property of the (sequential & parallel) trust propagation formulas. For the sake of simplicity, we adopt in the following the trust notions and the formulas of [8].

3.1 Matrix and monoids of trust

Definition 7. *Let $G = (V, E, \mathcal{T})$ be a trust graph, the* matrix of trust *of G, denoted by C, is the incidence matrix containing, for each node of the graph ,G the trust degrees of a node toward its neighbors, $C_{ij} =< td(i, j), dtd(i, j), ud(i, j) >$. When there is no edge between i and j, we choose $C_{ij} =< 0, 0, 1 >$ and, since every entity is fully confident in itself, we also choose for all i: $C_{ii} =< 1, 0, 0 >$.*

Definition 8. *Let $\mathcal{T}$ be the set $\mathcal{T} = \{< x, y, z > \in [0, 1]^3, x + y + z = 1\}$, equipped with two operations "+" and "." such that $\forall < a, b, u >, < c, d, v > \in \mathcal{T}$ we have: $< a, b, u > . < c, d, v > =< ac + bd, ad + bc, 1 - ac - ad - bd - bc >$, and $< a, b, u > + < c, d, v > =< 1 - (1 - a)(1 - c), bd, (1 - a)(1 - c) - bd >$. We define as the* monoids of trust *the monoids $(\mathcal{T}, +, < 0, 1, 0 >)$ and $(\mathcal{T}, ., < 1, 0, 0 >)$.*

$< 0, 0, 1 >$ is the absorbing element of "." in $\mathcal{T}$. This justifies a posteriori our choice of representation for the absence of an edge between two nodes in definition 7.

We can also see that the set $\mathcal{T}$ corresponds to trust degrees $< td, dtd, ud >$. In addition, the operations "." and "+" represent respectively the sequential and parallel aggregations of trust, denoted f and g in definitions 4 and 5. Finally, note that "." is *not* distributive over "+".

3.2 d-aggregation of trust

Definition 9. ***(d-aggregation of trust)*** *For $d \in \mathbb{N}$, the* d-aggregation of trust *between two nodes A and B, in an acyclic trust graph, is the trust evaluation over* all paths of length at most d, *connecting A to B. It is denoted $d\text{-}agg_{A,B}$.*

Definition 10. ***(Trust vectors product)*** *Consider the directed trust graph $G = (V, E, \mathcal{T})$ with trust matrix C. Let $\overrightarrow{C_{i*}}$ be the i-th row vector and $\overrightarrow{C_{*j}}$ be the j-th column vector. We define the product of $\overrightarrow{C_{i*}}$ by $\overrightarrow{C_{*j}}$ in the set $\mathcal{T}$ to be:*

$$\overrightarrow{C_{i*}}.\overrightarrow{C_{*j}} = \sum_{k \in V}^{k \neq j} C_{ik}.C_{kj}$$

Note that $C_{ii} = C_{jj} =< 1, 0, 0 >$ is the neutral element for ".". Therefore, our definition differs from the classical dot product as we have removed one of the $C_{ij} = C_{ii} \cdot C_{ij} = C_{ij} \cdot C_{jj}$, but then it matches the 2-aggregation:

Lemma 1. *The product $\overrightarrow{C_{i*}}.\overrightarrow{C_{*j}}$ is the 2-aggregated trust between i and j.*

Proof. We prove first that $C_{ik}.C_{kj}$ is the sequential aggregation of trust between i and j throughout all the paths (of length$\leq$ 2) $i \to k \to j$ with $k \in V$. Let k be an entity in the network there are two cases: whether k is one of the boundaries of the path or not. The first case is: $k = i$ or $k = j$:

- if $k = i$, then from the trust matrix definition 7, $C_{ii} =< 1, 0, 0 > \forall i$, thus we have $C_{ik}.C_{kj} = C_{ii}.C_{ij} =< 1, 0, 0 > .C_{ij} = C_{ij}$.

- if $k = j$, then similarly $C_{ik}.C_{kj} = C_{ij}.C_{jj} = C_{ij}. < 1, 0, 0 >= C_{ij}$

Therefore $C_{ik}.C_{kj}$ corresponds to the [sequential aggregation of] trust between i and j throughout the path (i, j) of length 1. This is why in the product, we added the constraint $k \neq j$ in the sum to avoid taking C_{ij} twice into account. Now the second case is: $k \neq i$ and $k \neq j$:

- if k belongs to a path of length 2 connecting i to j, then: i trusts k with degree $C_{ik} \neq < 0, 0, 1 >$, and k trusts j with degree $C_{kj} \neq < 0, 0, 1 >$. From definition 4, $C_{ik}.C_{kj}$ corresponds to the sequential aggregation of trust between i and i throughout the path $i \to k \to j$.

- If there is no path of length 2 between i and j containing k, then we have $C_{ik} =< 0, 0, 1 >$ or $C_{kj} =< 0, 0, 1 >$, and thus $C_{ik}.C_{kj} =< 0, 0, 1 >$ is also the aggregation of trust between i and j on the path traversing the node k.

Finally, we can deduce that $\overrightarrow{C_{i*}}.\overrightarrow{C_{*j}} = \sum_{k \in V}^{k \neq j} C_{ik}.C_{kj}$ corresponds to the parallel aggregation of trust between i and j using all paths of length ≤ 2, which is the 2-aggregated trust between i and j. Note that the latter is equivalent to $\overrightarrow{C_{i*}}.\overrightarrow{C_{*j}} = \sum_{k \in Pred.(j) \setminus \{i\}} C_{ik}.C_{kj} + C_{ij}$. □

Definition 11. ***(Trust matrix product)*** *Let $C_{(ij)}$ and $M_{(ij)}$ be two trust matrices. We define the matrix product $N = C * M$ by: $\forall i, j \in \{1..n\}$*

$$N_{ij} = \begin{cases} \overrightarrow{C_{i*}}.\overrightarrow{M_{*j}} = \sum_{k \in V}^{k \neq j} C_{ik}.M_{kj} & \text{if } i \neq j \\ < 1, 0, 0 > & \text{otherwise} \end{cases}$$

Lemma 2. *Let (C_{ij}) be the trust matrix of a network of entities, whose elements belong to a trust graph G. The matrix M defined by: $M = C^2 = C*C$ represents the 2-aggregated trust between all entities pairs.*

Proof. We have: $\forall i, j \in \{1..n\}$ $M_{ij} = \begin{cases} \overrightarrow{C_{i*}}.\overrightarrow{C_{*j}} = \sum_{k \in V}^{k \neq j} C_{ik}.C_{kj} & \text{if } i \neq j \\ < 1, 0, 0 > & \text{otherwise} \end{cases}$.
If $i = j$: then $M_{ii} =< 1, 0, 0 >$ denotes that i has a total trust on itself. Otherwise, if $i \neq j$: according to lemma 1, M_{ij} the 2-aggregated trust between i and j. □

Now, according to definition 6 of the trust evaluation in a network, we can generalize lemma 2 to evaluate trust using all paths of a given length:

Theorem 1. *Let $G = (V, E, \mathcal{T})$ be an acyclic trust graph with matrix of trust C. Then C^d represents the d-aggregated trust between all entity pairs in V.*

Proof. We proceed by induction. Let $HR(d)$ be the hypothesis that C^d represents the d-aggregated trust between all entity pairs in V. Then $HR(2)$ is true from lemma 2. Now let us suppose that $HR(d)$ is true. First if $i = j$, then $(d+1)\text{-agg}_{i,i} =< 1,0,0 >= C_{i,i}^{d+1}$ from definition 11. Second, definition 6, gives:

$$\begin{aligned}
(d+1)\text{-agg}_{i,j} &= \sum_{k \in Pred.(j)\setminus\{i\}} d\text{-agg}_{ik}.C_{kj} + C_{ij} && \text{by def. 6} \\
&= \sum_{k \in Pred.(j)\setminus\{i\}} C_{ik}^{d}.C_{kj} + C_{ij} && \text{by } HR(d) \\
&= \sum_{k \in Pred.(j)} C_{ik}^{d}.C_{kj} && \text{since } C_{ii} =< 1,0,0 > \\
&= \sum_{k \in V}^{k \neq j} C_{ik}^{d}.C_{kj} && \text{if } (ik) \notin E,\ C_{ik} =< 0,0,1 >
\end{aligned}$$

Overall, $HR(d+1)$ is proven and induction proves the theorem. □

From this theorem, we immediately have that in an acyclic graph the matrix powers must converge.

Corollary 1. *Let $G = (V, E, \mathcal{T})$ be an acyclic trust graph with trust matrix C. The matrix powers of C converges to a matrix C^{ℓ}, where ℓ is the size of the longest path in G.*

For the proof of this corollary, we need the following lemma.

Lemma 3. *Let $\lambda_{i,j}$ be the length of the largest path between i and j. Then either $\lambda_{i,j} = 1$ or $\lambda_{i,j} = max_{k \in Pred.(j)\setminus\{i\}}(\lambda_{i,k}) + 1$.*

Proof of corollary 1. We prove the result by induction and consider the hypothesis $HR(d)$ with $d \geq 1$:

$$\forall i, j \in V, \text{ such that } \lambda_{i,j} \leq d \text{ and } \forall t \geq \lambda_{ij}, \text{ then } C_{ij}^{t} = C_{ij}^{\lambda_{i,j}}$$

We first prove the hypothesis for $d = 1$: Let i and j be such that the longest path between them is of length 1. This means that in this acyclic directed graph, there is only one path between i and j, the edge $i \to j$. Now from definition 6, we have that $\forall t, C_{ij}^{t} = \sum_{k \in Pred.(j)\setminus\{i\}} C_{ik}^{t-1}.C_{kj} + C_{ij}$. However, $Pred.(j) \setminus \{i\} = \emptyset$ so that $C_{ij}^{t} = C_{ij}$, for all t. This proves $HR(1)$.

Now suppose that $HR(d)$ is true. Let i and j be two vertices in $G(V, E, \mathcal{T})$. We have two cases. First case: $\lambda_{ij} \leq d$. Then $\lambda_{i,j} \leq d + 1$ and from the induction hypothesis, we have that $C_{ij}^{t} = C_{ij}^{\lambda_{ij}} \forall t \geq \lambda_{ij}$. Therefore $HR(d+1)$ is true for i and j. Second case: $\lambda_{ij} = d + 1 \geq 2$. Then we have $\forall u \geq 0$ $C_{ij}^{d+1+u} = \sum_{k \in Pred.(j)} C_{ik}^{d+u}.C_{kj}$. Now, from lemma 3, the maximum length of any path between i and a predecessor of j is $\lambda_{i,j} - 1 = d$. Therefore, from the induction hypothesis, we have that $C_{ik}^{d+u} = C_{ik}^{\lambda_{ik}} = C_{ik}^{d}$ for all $k \in Pred.(j)$. Then $C_{ij}^{d+1+u} = \sum_{k \in Pred.(j)} C_{ik}^{d}.C_{kj} = C_{ij}^{d+1}$ which proves the induction and thus the corollary. □

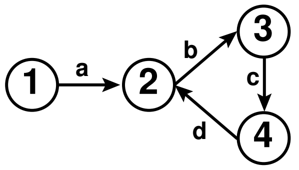
From the latter corollary, we now have an algorithm to compute the trust evaluation between all the nodes in an acyclic trust network: perform the trust matrix powering with the monoids laws up to the longest path in the graph.

Theorem 2. *Let (C_{ij}) be the trust matrix corresponding to an acyclic graph with n vertices and φ edges whose longest path is of size ℓ. The complexity of the evaluation of the aggregated trust between all entity pairs represented by this trust matrix is bounded by $O(n \cdot \varphi \cdot \ell)$ operations.*

Proof. C is sparse with φ non zero element. Thus multiplying C by a vector requires $O(\varphi)$ operations and computing $C \times C^i$ requires $O(n\varphi)$ operations. Then, theorem 1 shows that C^j for $j \geq \ell$ is the j-aggregated trust between any entity pair. Finally, corollary 1 shows that $C^j = C^\ell$ as soon as $j \geq \ell$. $\square$

3.3 Evaluation of trust in the presence of cycles

In the presence of cycles in a network, the matrix powers algorithm reevaluates indefinitely the trust degrees between the nodes in a cycle. This implies that the algorithm will converge finally to the maximal trust degree 1.



1	a	0	0
0	1	b	0
0	0	1	c
0	d	0	1

1	a	a.b	0
0	1	b	b.c
0	c.d	1	c
0	d	d.b	1

1	a	a.b	a.b.c
0	1	b	b.c
0	c.d	1	c
0	d	d.b	1

1	a+ a.b.c.d	a.b	a.b.c
0	1	b	b.c
0	c.d	1	c
0	d	d.b	1

1	a + a.b.c.d	(a+a.b.c.d).b	a.b.c
0	1	b	b.c
0	c.d	1	c
0	d	d.b	1

Figure 3: Graph with one cycle and its trust matrix C with C^2, C^3, C^4 and C^5.

Consider the graph of figure 3, with a, b, c, d the trust degrees corresponding to the links $1 \to 2 \to 3 \to 4 \to 2$. Its trust matrix C and applications of the matrix powers algorithm on this matrix are shown on figure 3. For instance, the value $C^5_{1,3} = 0 + C^4_{1,2}C_{2,3} + 0 = (C_{1,2} + C^3_{1,4}C_{4,2})C_{2,3} = (C_{1,2} + (C^2_{1,3}C_{3,4})C_{4,2})C_{2,3} = (C_{1,2} + (C_{1,2}C_{2,3})C_{3,4})C_{4,2})C_{2,3} = (a + a.b.c.d).b$, corresponds to the aggregation on the paths $1 \to 2 \to 3$ and $1 \to 2 \to 3 \to 4 \to 2 \to 3$ linking 1 to 3. If we continue iterations for $n > 5$, we find that the algorithm re-evaluates the trust on the loop $3 \to 4 \to 2 \to 3$ infinitely.

To solve this issue, we propose to change the matrix multiplication procedure, so that *each path will be used only once in the assessment of a trust relationship.* For this, we use a memory matrix R_{ij}. This stores, for each pair of nodes, all edges traversed to evaluate their trust degree. Only the paths containing an edge not already traversed to evaluate the trust degree are taken into account at the novel iteration. Therefore, the computation of C^{ℓ}_{ij} for $n \geq 1$, becomes:

Algorithm 2 Matrix powers for generic network graphs

Input An $n \times n$ matrix of trust C of a generic directed trust graph.
Output Global trust in the network.

```
1: $\ell = 2$;
2: **Repeat**
3:   **For all** $(i \neq j) \in [1..n]^2$ **do**
4:     $C^{\ell}_{ij} =< 0,0,1 >$; $R^{\ell}_{ij} = \emptyset$;
5:     **For** $k = 1$ **to** $n$ **do**
6:       $t = C^{\ell-1}_{ik}.C_{kj}$;
7:       **If** $(t \neq< 0,0,1 >)$ **then**
8:         $C^{\ell}_{ij} + = t$;
9:         $R^{\ell}_{ij} = R^{\ell}_{ij} \bigcup R^{\ell-1}_{ik} \bigcup (k \to j)$; // using a sorted set union
10:      **End If**
11:    **End For**
12:    **If** $\left(\#R^{\ell}_{ij} \subset \#R^{\ell-1}_{ij}\right)$ **then** $C^{\ell}_{ij} = C^{\ell-1}_{ij}$;$R^{\ell}_{ij} = R^{\ell-1}_{ij}$; **End If**
13:  **End For**
14: **Until** $C^{\ell} == C^{\ell-1}$; $++\ell$;
15: **return** $C^{\ell}$;
```

Theorem 3. *Let C be the trust matrix corresponding to a generic trust graph. Algorithm 2 converges to the matrix C^{ℓ} where ℓ is the longest* acyclic path *between vertices.*

Proof. Let C^{ℓ} be the evaluation of the ℓ-aggregated trust between all entity pairs after ℓ iterations where ℓ is the longest acyclic path between vertices. At this stage, for each pair i, j, all the edges belonging to a path between i and j will be marked in R^{ℓ}_{ij}. Therefore, no new $t = C^{\ell}_{ik}.C_{kj}$ will be added to $C^{\ell+1}_{ij}$. Conversely, at iteration $x < \ell$, if there exist an acyclic path between a pair i, j of length greater than x, then it means that there exists at least one edge e not yet considered on a sub-path from, say, u to v, of length x: $i \dashrightarrow u \dashrightarrow \xrightarrow{e} \dashrightarrow v \dashrightarrow j$. Then R^{x}_{uv} will be different from R^{x-1}_{uv} and so will be C^{x}_{uv} from C^{x-1}_{uv}. □

Theorem 4. *Let C be the trust matrix corresponding to a generic trust graph with n vertices and φ edges whose longest path between vertices is of size ℓ. The complexity of the global evaluation of all the paths between any entity is bounded by $O(n^3 \cdot \varphi \cdot \ell)$ operations.*

Proof. Using algorithm 2, we see that the triple loop induces n^3 monoid operations and n^3 merge of the sorted sets of edges. A merge of sorted sets is linear in the number of edges, φ. Then the overall iteration is performed at most ℓ times from theorem 3. □

By applying the new algorithm on the example of figure 3, we still obtain $C^5_{1,3} = (a + a.b.c.d).b$, but now $R^5_{1,3} = \{a, b, c, d\}$ and thus no more contribution can be added to $C^{5+i}_{1,3}$.

A first naive dense implementation of this algorithm took less than 4 seconds to perform the first iteration (C^2) on the graph of section 2 with 1000 vertices and $250k$ edges.

3.4 Bounded evaluation of trust

In practice, the evaluation of trust between two nodes A and B need not consider all trust paths connecting A to B for two reasons:

- First, the mitigation is one of the trust properties, i.e. the trust throughout trust paths decreases with the length of the latter. Therefore after a certain length L, the trust on paths becomes weak and thus should have a low contribution in improving the trust degree after their parallel aggregation.

- Second, if at some iteration $n \geq 1$, we already obtained a high trust degree, then contributions of other paths will only be minor.

Therefore, it is possible to use the matrix powers algorithm with less iterations and e.g. a threshold for the trust degree, in order to compute fast a good approximation of the trust in a network.

4 Distributed Trust and Reputation for public key architectures

The reputation can be defined as in [13]: "*a peer's belief in another peer's capabilities, honesty and reliability based on the other peers recommendations.*" Currently, the reputation is implemented in several areas: e-commerce , mailing (combating spams), search engines (Pages classification), P2P networks, ... Reputation can for instance be used to help peers distinguish *good from bad partners.*

However, the notions of trust and reputation are usually treated separately. Yet these two concepts are complementary are both necessary for a better quantification of the credibility of an entity.

On the one hand, it is necessary to have at least one trust path between two entities to evaluate their trust degree. This cannot always be guaranteed in large networks. There, the reputation can give a significant indication that will allow users to take the decision to communicate (or not) with other peers.

On the other hand, a low degree of reputation cannot be conclusive on the credibility of an entity. Indeed, reputation depends on the number of incoming trust relationships and this may discriminate the least "popular" entities. In this case, the trust degree is more significant.

There exist several reputation evaluation systems like e.g. EigenTrust [10], inspired from Google's PageRank [11] or the Spreading Activation Model for trust propagation [14], etc. An important advantage of reputation systems is their performance. They are fast compared to the binary trust evaluation on a network, usually in time quadratic in the size of the network.

In the following, we propose a model combining the trust and the reputation concepts, for an efficient evaluation of trust in PKI architectures, i.e. with complexity cubic in the size of the network.

The first question is between a centralized and a distributed model of evaluation. To implement a centralized trust management model, we would need a new "trusted authority" (TA) which would assess trust in all PKI architectures. This is somewhat the case e.g. in the DNSsec specifications. Its role here would be to retrieve the trust degrees expressed by all CAs and to evaluate the trust and reputation degrees in the network. But then TA would need to have a global vision of the network and to estimate with high accuracy the trust degrees between entities.

Moreover, the reliability of this centralized model is based entirely on the reliability of the TA. This might not be applicable to very large network like internet but more suited to small "local" communities of CAs.

Another approach is to use a distributed management model, where the entities must contact others to share some trust degrees. This will enable each entity to evaluate the trust in its neighborhood. On the one hand, this can be applied to large networks, while preserving for each entity a low computational cost. On the other hand, each entity might have only a limited view of the whole network.

Now, in this setting it is also possible to distribute even the computation of the trust matrix: each entity would be responsible of the computation of a sub-matrix of the global network. Then the entity could receive some other (potentially overlapping) sub-matrices, signed by trusted entities. These trust degrees could be expressed in the certificates, even as a shared secret [2].

Our model can be applied to any PKI system (like cross-certified PKIs or the PGP web of trust), which consists in a number of entities with certificates/keys, and in which any entity may sign other entities' certificates/keys.

This cross-certification/key-signature supposes at least the verification of a policy along the certification path from the signatory to the owner of the signed certificate. Thus, each entity may express its trust degrees in the certificates/keys it has to create/sign.

Now, due to the difficulty of expressing precisely a trust degree a first rating could use a scale from 0 to 10 to normalize the triplet (trust, distrust, uncertainty). As shown for example on figure 4, one can evaluate a trust degree to 6/10, usually a smaller value for the known false emissions of the trustee (often 0/10 or here 1/10) and then the uncertainty is deduced as

$1 - trust - distrust = 3/10$.

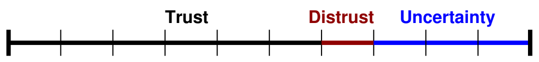


Figure 4: Example of a scale of trust

4.0.1 Cross-certified PKI architectures

In the case of cross-certified PKI architectures, the CAs play the main role. In the context of this article, we assume that the relations $[CA \rightarrow users]$ are based on complete trust. In this case, only the inter-CAs relationships are evaluated. Each CA creates an initial trust matrix from its certificate store and saves it locally. This matrix corresponds to the sub-graph of the CAs neighborhood. A network discovery mechanism should then be established to expand the trust sub-graph and to have a broader view on the network. Then the CA evaluates the trust and reputation using the matrix powers algorithm of section 3 and a reputation evaluation scheme. Then it can also decide to forward some of this information to its users, via e.g. its SCVP services (Server-based Certificate Validation Protocol), in response to their certificate validation requests.

4.0.2 PGP Web Of Trust

In the case of PGP networks, the same rating system could replace the actual system (full trust, marginally trusted, no trust). This will allow to assess more precisely the trust degrees between users. Each user creates its own trust matrix, which will be initialized from certificates (public keys) in the key-ring. A network discovery mechanism could also be established to expand the local network of trust. Finally, trust and reputation are assessed through the trust matrix.

For instance, the PGP client settings: *COMPLETS_NEEDED*, *MARGINALS_NEEDED* which are used to compute the required number of signatures generated by keys with full or marginal trust could be replaced by: *MINIMAL_TRUST* and *MINIMAL_REPUTATION*, representing the trust and reputation degrees needed to validate a public key. The values of these parameters will depend of course on the used trust propagation system and personal policies.

5 Conclusion

The actual public-key infrastructure models assume that the relationships between the PKI entities are based on an absolute trust. However, several risks when using PKI procedures are related to these assumptions. In this article we introduce a simple distributed trust model in order to quantify and manage the trust in cross-certified PKIs and in the PGP web of trust. We use the formal semantics based calculus of trust introduced by [7, 8] and apply it to the PKIs.

We have reduced the evaluation of trust between entities of a DAG network to linear algebra. This gives a polynomial algorithm to asses the global trust evaluation in a network. Moreover, depending on the sparsity of the considered graphs this enables to use adapted linear algebra methods. Also the linear algebra algorithm decomposes the evaluation into converging iterations which could be stopped before exact converge in order to get a good approximation faster. Finally this enabled us also to generalize the trust evaluation to any directed graph, still with a polynomial complexity.

Overall, our model combines the reputation and the trust notions to give a precise indication about the credibility of the PKI entities.

Further improvement includes a dedicated Network Discovery Mechanism, used to expand the trust sub-graph and to guaranty the safety in the trust model.

Also the trust degrees could be a sensitive information. Therefore, the join use of trust matrices and homomorphic cryptosystems enabling a private computation of shared secret would be useful.

References

[1] H. Adkins. An update on attempted man-in-the-middle attacks. Technical report, Google Online Security Blog, Aug. 2011. `http://googleonlinesecurity.blogspot.com/2011/08/update-on-attempted-man-in-middle.html`.

[2] S. Dolev, N. Gilboa, and M. Kopeetsky. Computing multi-party trust privately: in o(n) time units sending one (possibly large) message at a time. In *Proceedings of the 2010 ACM Symposium on Applied Computing*, SAC '10, pages 1460–1465, New York, NY, USA, 2010. ACM.

[3] C. Ellison and B. Schneier. Ten risks of PKI: What you're not being told about Public Key Infrastructure. *Computer Security Journal*, 16(1):1–7, 2000. `http://www.counterpane.com/pki-risks.pdf`.

[4] E. Estrada and N. Hatano. Communicability graph and community structures in complex networks. *Applied Mathematics and Computation*, 214(2):500–511, 2009.

[5] F. Gomes. Security alert: Fraudulent digital certificates. Technical report, SANS Institute InfoSec Reading Room, June 2001. `http://www.sans.org/reading_room/whitepapers/certificates/security-alert-fraudulent-digital-certificates_679`.

[6] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins. Propagation of trust and distrust. In *Proceedings of the 13th international conference on World Wide Web*, WWW '04, pages 403–412, New York, NY, USA, 2004. ACM.

[7] J. Huang and D. Nicol. A calculus of trust and its application to pki and identity management. In *Proceedings of the 8th Symposium on Identity and Trust on the Internet*, IDtrust '09, pages 23–37, New York, NY, USA, 2009. ACM.

[8] J. Huang and D. Nicol. A formal-semantics-based calculus of trust. *IEEE Internet Computing*, 14:38–46, September 2010.

[9] A. Jøsang. Probabilistic logic under uncertainty. In *Proceedings of Computing: The Australian Theory Symposium (CATS'07)*, January 2007.

[10] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina. The eigentrust algorithm for reputation management in p2p networks. In *Proceedings of the 12th international conference on World Wide Web*, WWW '03, pages 640–651, New York, NY, USA, 2003. ACM.

[11] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, Nov. 1999.

[12] M. K. Reiter and S. G. Stubblebine. Resilient authentication using path independence. *IEEE Trans. Comput.*, 47:1351–1362, December 1998.

[13] Y. Wang and J. Vassileva. Trust and reputation model in peer-to-peer networks. In *Proceedings of the 3rd International Conference on Peer-to-Peer Computing*, P2P '03, pages 150–, Washington, DC, USA, 2003. IEEE Computer Society.

[14] C.-N. Ziegler and G. Lausen. Spreading activation models for trust propagation. In *Proceedings of the 2004 IEEE International Conference on e-Technology, e-Commerce and e-Service (EEE'04)*, EEE '04, pages 83–97, Washington, DC, USA, 2004. IEEE Computer Society.